TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta přírodovědně-humanitní
a pedagogická

Core problém v lineární aproximační úloze $Ax \approx b$ s jednou pravou stranou

Bakalářská práce

Studijní program: B1101 – Matematika
Studijní obor: 1101R016 – Matematika

Autor práce: **Filip Jágr**
Vedoucí práce: Martin Plešinger

Liberec 2015

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Science, Humanities
and Education

The core problem within a linear approximation problem $Ax \approx b$, with the single right-hand side

Bachelor thesis

Study programme: B1101 – Mathematics
Study branch: 1101R016 – Mathematics

Author: **Filip Jágr**
Supervisor: Martin Plešinger

Liberec 2015

Tento list nahraďte
originálem zadání.

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum:

Podpis:

Anotace

Tato bakalářská práce se zabývá problematikou řešení lineárních aproximačních úloh ve smyslu tzv. úplných nejmenších čtverců (TLS, z anglického total least squares).

V úvodu se seznámíme s velmi důležitým nástrojem, kterým je singulární rozklad (SVD, z anglického singular value decomposition). Dále v práci popíšeme možné přístupy k řešení lineárních aproximačních úloh. Nejprve stručně zopakujeme běžně známý přístup přeformulování úlohy na tzv. (klasický) problém nejmenších čtverců, neboli lineární regresi. V kontrastu k tomu zavedeme úplný problém nejmenších čtverců, neboli tzv. ortogonální regresi. Dokážeme, že ne vždy má lineární aproximační úloha řešení ve smyslu TLS. Tento důkaz provedeme pomocí ortogonální transformace původní úlohy, tj. změny báze souřadného systému, ve kterém je úloha zformulovaná, na blokově diagonální tvar. Tato transformace povede přímo k definici tzv. core problému. Zmíníme, že core problém lze zapsat ve dvou typických tvarech, pomocí SVD tvaru a pomocí bidiagonálního tvaru.

V závěrečné části popíšeme software vytvořený v prostředí MATLAB, který dokáže generovat pseudonáhodné lineární aproximační úlohy předepsaných rozměrů obsahující core problém předepsaných vlastností. Smyslem tohoto softwaru je vytvoření databáze úloh pro statistické testy. To však již není součástí této práce.

Klíčová slova:

singulární rozklad (SVD); (klasický) problém nejmenších čtverců (LS); úplný problém nejmenších čtverců (TLS); ortogonální regrese; ortogonální transformace; core problém; SVD tvar core problému; bidiagonální tvar core problému

Abstract

This bachelor thesis deals with solving linear approximation problems in the sense of the so-called total least squares (TLS).

In the beginning we introduce one of the most important tools, the singular value decomposition (SVD). Then we briefly describe possible approaches to solving such problem. First approach is the commonly known (ordinary) least squares formulation (or problem) also know as linear regression. We are particularly interested in the second approach, the total least squares formulation (or problem) also know as orthogonal regression. We show that the linear approximation problem may not have a solution in the sense TLS. This can be show using orthogonal transformations of the original problem, i.e., by changing coordinate systems in which the problem is formulated, so that the transformed problem has a block diagonal form. This transformation leads directly to the definition of the so-called core problem. We mention that the core problem can be written in two typical forms, in the SVD form and in the bidiagonal form.

In the final section, we describe a software tool that we developed in MATLAB, which can generate pseudo-random linear approximation problems with given dimensions and containing a core problem with prescribed properties. The aim of this tool is to assemble a database of such problems, which will be used for futher statistcal testing. This, however, is not part of this thesis.

Key words:

singular value decomposition (SVD); (ordinary) least squares problem (LS); total least squares problem (TLS); orthogonal regression; orthogonal transformation; core problem; SVD form of a core problem; bidiagonal form of a core problem

Poděkování

Chtěl bych poděkovat Martinu Plešingerovi za vedení mé bakalářské práce, cenné rady, trpělivost a ochotu, kterou mi věnoval v průběhu zpracování této práce.

Obsah

Seznam obrázků

Značení a základní pojmy

Matice a vektory

$A \in \mathbb{R}^{n\times m}$ $(\mathbb{C}^{n\times m})$	reálná (komplexní) matice m-krát-n s prvky $a_{i,j}$
A^T	transponovaná matice k matici A
$\overline{A}$	komplexně sdružená matice k matici A
$A^H = \overline{A}^T$	hermitovsky sdružená matice k matici A
$\mathrm{rank}(A)$	hodnost matice definovaná jako počet lineárně nezávislých řádků, respektive sloupců matice A
$Q^T = Q^{-1} \in \mathbb{R}^{n\times n}$	ortogonální matice
$Q^H = Q^{-1} \in \mathbb{C}^{n\times n}$	unitární matice
$x \in \mathbb{R}^n$ $(\mathbb{C}^n)$	reálný (komplexní) vektor s délkou n a prvky x_j
$\|x\|$	eukleidovská norma vektoru, $\|x\| = (\sum_j \|x_j\|^2)^{1/2}$
$\langle x, y\rangle$	standardní skalární součin, $\langle x, y\rangle = y^H x = \sum_j x_j \overline{y}_j$
$x \perp y$	vektory x a y jsou ortogonální, tj. $\langle x, y\rangle = 0$
$\|A\|_2$	spektrální norma matice, $\|A\|_2 = \max_{\|x\|=1} \|Ax\|$
$\|A\|_F$	Frobeniova norma matice, $\|A\|_F = (\sum_i \sum_j \|a_{i,j}\|^2)^{1/2}$
$\Sigma(A)$	množina všech singulárních čísel matice A včetně násobností
$\mathcal{N}(A)$	jádro matice A
$\mathcal{R}(A)$	obor hodnot matice A
$\dim(\mathcal{V})$	dimenze lineárního vektorového prostoru $\mathcal{V}$
$\mathcal{V}^\perp$	ortogonální doplněk podprostoru $\mathcal{V}$
$\mathcal{V} \perp \mathcal{W}$	podprostory $\mathcal{V}$ a $\mathcal{W}$ jsou ortogonální, tj. $\langle x, y\rangle = 0$, $\forall x \in \mathcal{V}$ a $\forall y \in \mathcal{W}$
$\mathcal{V} \oplus \mathcal{W}$	je direktní součet podprostorů $\mathcal{V}$ a $\mathcal{W}$
$\mathrm{span}(q_1, \ldots, q_m)$	prostor generovaný vektory $q_1, \ldots, q_m$
$\min \mathcal{M}$, $\mathcal{M} \subset \mathbb{R}$	minimum z množiny $\mathcal{M}$
$\|\mathcal{M}\|$	počet prvků konečné množiny $\mathcal{M}$

Terminologie aproximačích úloh

$Ax \approx b$	aproximační úloha s maticí $A \in \mathbb{R}^{n\times m}$ a vektorem pravé strany $b \in \mathbb{R}^n$, $b \notin \mathcal{R}(A)$
$A_{11}x_1 \approx b_1$	core problém v úloze $Ax \approx b$, $A_{11} \in \mathbb{R}^{\overline{n}\times\overline{m}}$, $b_1 \in \mathbb{R}^{\overline{n}}$
A	systémová matice (matice modelu)
b	pravá strana (vektor pozorování)
$[b, A]$	rozšířená matice (matice dat)

Použité zkratky

SVD	singulární rozklad matice, $A = U\Sigma V^H$ (singular value decomposition)
TLS	úplný problém nejmenších čtverců (total least squares problem)

Úvod

Ústředním bodem této práce bude aproximační problém

$$Ax \approx b,$$

kde $A \in \mathbb{R}^{n\times m}$ je daná matice, reprezentující např. nějaký model a pravá strana $b \in \mathbb{R}^n$ je daný vektor pozorování, který může být přirozeně zatížený chybami, např. chybami měření. Obecně tedy vektor b nemusí ležet v oboru hodnot $\mathcal{R}(A)$ matice A. Proto budeme dále striktně předpokládat

$$b \notin \mathcal{R}(A),$$

v opačném případě by byl aproximační problém de-facto soustavou rovnic $Ax = b$ a měl by řešení v klasickém slova smyslu. Protože ale problém $Ax \approx b$, $b \notin \mathcal{R}(A)$ řešení nemá (neexistuje vektor x takový, aby nastala rovnost), musíme přistoupit k nějakému náhradnímu postupu. Typicky hledáme vektor x takový, abychom minimalizovali nějaký předepsaný funkcionál. Často se v takovém případě používá nějaká varianta metody nejmenších čtverců. V této práci se budeme zabývat tzv. *úplným problémem nejmenších čtverců* (TLS, z anglického *total least squares*), který hledá nejmenší matici E a vektor r, tj.

$$\min \|[r, E]\|_F \quad \text{tak, že} \quad (b + r) \in \mathcal{R}(A + E),$$

kde $\|\cdot\|_F$ značí tzv. Frobeniovu normu matice, tj. odmocninu ze součtu kvadrátů (absolutních hodnot) všech prvků matice. Pokud taková minimální E a r existují, pak libovolný vektor x splňující $(A + E)x = (b + r)$ nazýváme *TLS řešení* původní úlohy $Ax \approx b$. Tato úloha (na rozdíl od klasického problému nejmenších čtverců, kde hledáme pouze minimální opravu r pravé strany b) obecně nemá řešení pro dané A a b, jak již v roce 1980 ukázali G. H. Golub a C. F. Van Loan v článku [8]. Řadu praktických aplikací úplného problému nejmenších čtverců nalezneme např. v knihách [19], [20].

Další významný posun v analýze TLS problému byl publikován v roce 1991 v knize [18] J. Vandewalle a S. Van Huffel (vycházející z dizertační práce S. Van Huffel [15]). Zde se mimo jiné zavádí klasifikace problémů $Ax \approx b$ na tzv. *generické (generic)* a *negenerické (nongeneric) problémy*, přičemž první z nich *vždy mají TLS řešení* (ne nutně jednoznačné), zatímco negenerické problémy TLS řešení nemají. Termín „generický/negenerický problém“ ovšem není nejšťastněji zvolen. Pojem „generic“ lze z angličtiny přeložit jako „běžný“ a jeho antonymum „nongeneric“

pak např. jako „vzácný“. Tato terminologie byla pravděpodobně motivována tím, že se běžných výpočetních úlohách skutečně vyskytovaly zpravidla jen generické problémy.

Nejjednodušší případ TLS problému, který je v literatuře také nejčastěji zmiňován, navíc pracuje s maticí A plné sloupcové hodnosti, tj. $\operatorname{rank}(A) = m$. Spojení těchto dvou jevů nás může snadno svést k následující myšlence: „Má-li matice A lineárně nezávislé sloupce, pak má problém $Ax \approx b$ TLS řešení,“ která však *není pravdivá*. S. Van Huffel v článku [17] na str. 547 opatrně píše: „*These conditions* (podmínky postačující pro existenci TLS řešení) *are generically satisfied provided A is of full rank and the set* $Ax \approx b$ *is not too conflicting.*[1]“ V dalších pracech můžeme nalézt i méně opatrné formulace, např.: „*If we suppose that* $[b, A]$ *has full column rank, this condition* (podmínka postačující pro existenci TLS řešení) *is generally satisfied*,“ viz [5, str. 54]; nebo dokonce přímo nepravdivé, např.: „*In our case, when the matrix A has full rank and* $n > m$*, the solution of the TLS problem always exists*,“ viz [6, str. 37].

Nepravdivost výše zmíněného tvrzení, tedy fakt, že existují problémy $Ax \approx b$, $b \notin \mathcal{R}(A)$, $\operatorname{rank}(A) = m$, které nemají TLS řešení, lze nejsnáze ukázat pomocí tzv. *core problému* zavedeného C. C. Paigem a Z. Strakošem v článku [12]. Užitím core problému lze dokonce takové problémy snadno konstruovat. Cílem této práce je vytvořit software v prostředí MATLAB, který bude generovat pseudonáhodné aproximační problémy $Ax \approx b$ (tedy matice A a vektory b), předepsaných rozměrů, obsahující core problém předepsaných rozměrů a mající další předepsané vlastnosti. Tím budeme schopni generovat velké množství aproximačních úloh s maticemi plné sloupcové hodnosti a nemající TLS řešení.

Dalším krokem, který by měl navazovat, ale *není již součástí této práce*, bude vytvořit několik datových souborů s problémy $Ax \approx b$ (každý soubor charakterizovaný parametry, podle kterých byl vytvořen) dostatečně velkých pro následující statistickou analýzu. Konkrétně by bylo zajímavé naučit se generovat (tj. odladit parametry) problémy $Ax \approx b$, jejichž prvky (tj. prvky jejich matic A a pravých stran b) se budou chovat jako *nezávislé, identicky distribuované náhodné proměnné*, matice A by měly lineárně nezávislé sloupce, a problémy by přitom neměly TLS řešení. Zajímavý by byl i negativní výsledek, tedy pokud by se takové problémy generovat nepodařilo. Naznačovalo by to, že existenci core problému uvnitř aproximačního problému by mohlo být možné detekovat statistickými metodami.

Struktura této práce je následující. Po stručném úvodu, v kapitole 1 zavedeme tzv. singulární rozklad matice, jehož zavedení motivujeme spektrálním rozkladem. V závěru kapitoly ukážeme, jak lze danou matici aproximovat maticí nižší hodnosti. V kapitole 2 se budeme věnovat úplnému problému nejmenších čtverců (TLS), který motivujeme jednoduchým fyzikálním příkladem na elektrickou vodivost a tzv. klasickým problémem nejmenších čtverců. V závěru kapitoly zformulujeme postačující podmínku pro existenci řešení aproximační úlohy ve smyslu TLS. V kapitole 3 se

[1] Článek [17] je také k dispozici v preprintu [16], kde je citovaná věta na str. 9. V článku [17] i v jeho preprintu [16] se místo $Ax \approx b$ používá značení $X\beta \approx y$ obvyklejší ve statistické literatuře. Zde je značení obměněno z důvodu konzistence textu. Obdobně je značení obměněno v obou následujících větách citovaných z článků [5] a [6].

budeme věnovat tzv. core problému nástroji, který vždy nalezne řešení aproximační úlohy za předpokladu, že řešení existuje. V kapitole se budeme také věnovat různým tvarům core problému a sice tzv. SVD tvaru core problému a tzv. Bidiagonálnímu tvaru core problému. V kapitole 4 se budeme věnovat popisu naprogramovaných programů `cpSVD` a `cpBDG`, které generují core problém v SVD respektive v bidiagonálním tvaru a také programu `aproxpb`, který bude generovat aproximační úlohu $Ax \approx b$ s core problémem.

1 Singulární rozklad

V následující kapitole se budeme věnovat singulárnímu rozkladu (SVD). Ve výkladu SVD budeme vycházet ze skript [2], respektive z knih [21], [4], [9]. SVD je silným nástrojem pro výpočet mnoha vlastností matic. Pomocí SVD můžeme spočítat například hodnost, spektrální nebo Frobeniovu normu matice, pseudoinverzní matici, bázi nulového prostoru a oboru hodnot matice a řadu dalších vlastností. Za variabilitu a sílu SVD nicméně musíme počítat s vyššími výpočetními nároky.

Odvození singulárního rozkladu budeme motivovat spektrálním rozkladem symetrické matice.

1.1 Spektrální rozklad symetrické matice

Spektrální rozklad symetrické semidefinitní matice popisuje následující věta.

Věta 1. *Nechť $S \in \mathbb{R}^{n\times n}$ je symetrická pozitivně semidefinitní matice s hodností* $\operatorname{rank}(S) = r$. *Potom existuje ortogonální matice $Q \in \mathbb{R}^{n\times n}$ tak, že*

$$D = Q^T S Q \tag{1.1}$$

je diagonální matice s vlastními čísly matice S na diagonále. Ta jsou reálná a seřazená tak, že platí

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r > \lambda_{r+1} = \ldots = \lambda_n = 0.$$

Důkaz. Schurova věta [2, věta 2.2] říká, že pro každou matici S existuje *unitární* matice $Q \in \mathbb{C}^{n\times n}$, $Q = Q^H = \overline{Q}^T$, taková, že

$$S = QRQ^H,$$

kde

$$R = Q^H S Q$$

je *komplexní* horní trojúhelníková matice s vlastními čísly matice S na diagonále seřazenými v libovolném pořadí. Důkaz bude veden ve třech krocích. V prvním kroku ukážeme, že vlastní čísla matice S jsou reálná. Dále ukážeme, že vlastní čísla jsou nezáporná. Nakonec ukážeme, že matici Q lze volit reálnou.

Krok 1: Je zřejmé, že

$$S^T = (QRQ^H)^T = QR^HQ^H.$$

Protože je matice S podle předpokladu věty symetrická, tedy $S = S^T$, platí

$$QRQ^H = QR^HQ^H.$$

Vynásobením obou stran rovnice maticí Q^H zleva získáme

$$RQ^H = R^HQ^H$$

dalším vynásobením obou stran této rovnice maticí Q zprava vidíme, že platí

$$R = R^H = \overline{R}^T,$$

tedy je patrné, že matice R musí být diagonální a reálná.

Krok 2: Protože matice S je pozitivně semidefinitní, tedy

$$x^TSx \geq 0$$

pro libovolné $x \in \mathbb{R}^n$, pak

$$q_j^TSq_j = q_j^T(q_j\lambda_j) = (q_j^Tq_j)\lambda_j = \lambda_j \geq 0$$

pro $j = 1, \ldots, n$. Protože platí

$$\text{rank}(S) = \text{rank}(QDQ^T)$$

a Q je regulární, potom

$$\text{rank}(S) = \text{rank}(D),$$

protože matice D je diagonální, t.j. $D = \text{diag}(\lambda_1, \lambda_2, ..., \lambda_n)$, a

$$\text{rank}(D) = r,$$

pak právě $n - r$ vlastních čísel matice A musí být rovno nule. Tedy zbývajících r vlastních čísel je kladných.

Krok 3: Vztah $S = QDQ^H$ lze přepsat ve tvaru

$$SQ = QD, \tag{1.2}$$

t.j.

$$Sq_j = q_j\lambda_j, \qquad j = 1, ..., n, \tag{1.3}$$

kde q_j je j-tý sloupec matice Q a λ_j je j-tý diagonální prvek matice D. Tedy q_j je vlastní vektor matice S odpovídající vlastnímu číslu λ_j, t.j.

$$(S - I\lambda_j)q_j = 0. \tag{1.4}$$

Protože matice $S - I\lambda_j$ je reálná má homogenní soustava (1.4) také reálné řešení. Tedy vlastní vektor q_j a celou matici Q lze volit reálné. □

1.2 Symetrická matice jako lineární zobrazení

Každá čtvercová matice $S \in \mathbb{R}^{n\times n}$ představuje lineární zobrazení z prostoru $\mathbb{R}^n$ zpět do prostoru $\mathbb{R}^n$. Toto zobrazení přiřadí obecnému vektoru $x \in \mathbb{R}^n$ vektor $Sx \in \mathbb{R}^n$, tedy

$$S : x \quad \longmapsto \quad y = Sx.$$

Připomeňme, že pro *libovolné* lineární zobrazení (a tedy i pro matici) lze zavést pojem *obor hodnot zobrazení* a *jádro zobrazení*. Nejprve zavedeme pojem obor hodnot.

Definice 1 (Obor hodnot matice)**.** Nechť $A \in \mathbb{R}^{n\times m}$ je obecně obdelníková matice. Množina

$$\mathcal{R}(A) = \{y\,;\, y = Ax, x \in \mathbb{R}^m\} \subseteq \mathbb{R}^n$$

se nazývá obor hodnot matice A.

Následující věta upřesní jakou strukturu má obor hodnot matice.

Věta 2. *Nechť $A \in \mathbb{R}^{n\times m}$ je obecně obdelníková matice. Množina $\mathcal{R}(A)$ z definice 1 je lineární vektorový prostor.*

Důkaz. Důkaz je jednoduchý. Uvažujme dva libovolné vektory z oboru hodnot matice, např. $y_1, y_2 \in \mathcal{R}(A)$, pak existují vektory x_1, x_2, pro které platí $y_1 = Ax_1$ a $y_2 = Ax_2$. Dále zvolme $x = \alpha x_1 + \beta x_2$, potom

$$Ax = A(\alpha x_1 + \beta x_2) = \alpha Ax_1 + \beta Ax_2 = \alpha y_1 + \beta y_2 = y \in \mathcal{R}(A).$$

□

Nyní zavedeme druhý z pojmů, tedy jádro matice.

Definice 2 (Jádro matice)**.** Nechť $A \in \mathbb{R}^{n\times m}$ je obecně obdelníková matice. Množina

$$\mathcal{N}(A) = \{x\,;\, Ax = 0\} \subseteq \mathbb{R}^m$$

se nazývá jádro matice A.

Stejně jako u oboru hodnot, následující věta upřesní strukturu jádra matice.

Věta 3. *Nechť $A \in \mathbb{R}^{n\times m}$ je obecně obdelníková matice. Množina $\mathcal{N}(A)$ z definice 2 je lineární vektorový prostor.*

Důkaz. Pro vektory platí $x_1, x_2 \in \mathcal{R}(A)$. Dále zvolme $x = \alpha x_1 + \beta x_2$, potom

$$Ax = A(\alpha x_1 + \beta x_2) = \alpha Ax_1 + \beta Ax_2 = \alpha 0 + \beta 0 = 0.$$

□

Protože matice S je symetrická pozitivně semidefinitní a podle věty 1 víme, že má n vlastních vektorů a navíc, že tyto vektory lze volit navzájem ortogonální. Tedy vlastní vektory q_j, $j = 1, \ldots, n$, matice S tvoří ortonormální bázi $\mathbb{R}^n$. Tedy ze vztahů (1.1) respektive (1.2) a (1.3) plyne

$$\begin{aligned} S : q_1 &\longmapsto \lambda_1 q_1 , \\ S : q_2 &\longmapsto \lambda_2 q_2 , \\ &\vdots \\ S : q_r &\longmapsto \lambda_r q_r , \\ S : \{q_{r+1}, \ldots, q_n\} &\longmapsto 0 . \end{aligned} \tag{1.5}$$

Libovolný vektor $x \in \mathbb{R}^n$ můžeme zapsat jako lineární kombinaci bázových vektorů q_j, $j = 1, \ldots, n$ následujícím způsobem

$$x = \sum_{j=1}^{n} \alpha_j q_j. \tag{1.6}$$

Čísla α_j jsou souřadnice vektoru x ve směru q_j. Pak

$$S : x \longmapsto \sum_{j=1}^{n} \alpha_j (\lambda_j q_j).$$

Z toho plyne

$$\begin{aligned} \mathcal{R}(S) &= \operatorname{span}(q_1, q_2, \ldots, q_r) = \mathcal{R}(S^T), \\ \mathcal{N}(S) &= \operatorname{span}(q_{r+1}, \ldots, q_n) = \mathcal{N}(S^T). \end{aligned} \tag{1.7}$$

Protože platí $Q^T Q = I$ vidíme, že pro $\forall y \in \mathcal{R}(S)$ a pro $\forall x \in \mathcal{N}(S)$ platí, že jejich skalární součin bude roven nule, t.j.

$$\langle x, y \rangle = 0.$$

Ze skalárního součinu plyne, že obor hodnot matice a jádro matice jsou na sebe kolmé, tedy

$$\mathcal{R}(S) \perp \mathcal{N}(S). \tag{1.8}$$

Vektor (1.6) lze také zapsat ve tvaru

$$x = x_R + x_N,$$

kde x_R a x_N jsou definovány následovně

$$x_R = \sum_{j=1}^{r} \alpha_j q_j \in \mathcal{R}(S),$$

$$x_N = \sum_{j=r+1}^{n} \alpha_j q_j \in \mathcal{N}(S).$$

Pak pro x_R a x_N plyne z (1.8), že

$$x_R \perp x_N.$$

Tuto vlastnost zapisujeme

$$\mathcal{N}(S) \oplus \mathcal{R}(S) = \mathbb{R}^n\,, \tag{1.9}$$

kde operace $\oplus$ se nazývá direktní součet podprostorů. V další kapitole se pokusíme získat popis analogický k (1.5) a (1.7) pro obecnou obdelníkovou matici.

1.3 Obecná matice jako lineární zobrazení

Protože dále budeme pracovat s obecnou obdelníkovou maticí je třeba vyjasnit následující. Ze základního přehledu přednášek z lineární algebry víme, že počet lineárně nezávislých řádků se rovná počtu lineárně nezávislých sloupců. Tedy pro obecnou obdelníkovou matici $A \in \mathbb{R}^{n\times m}$ platí

$$\operatorname{rank}(A) = \operatorname{rank}(A^T),$$

přičemž hodnost matice A je definována jako dimenze prostoru $\mathcal{R}(A)$, tedy

$$\operatorname{rank}(A) = \dim(\mathcal{R}(A)).$$

Zřejmě tedy platí

$$\dim(\mathcal{R}(A)) = \operatorname{rank}(A) = \operatorname{rank}(A^T) = \dim(\mathcal{R}(A^T)). \tag{1.10}$$

Platí následující věta zobecňující vztahy (1.8) a (1.9).

Věta 4. *Uvažujme obecnou obdelníkovou matici $A \in \mathbb{R}^{n\times m}$. Pro kterou platí*

$$\mathcal{N}(A) \oplus \mathcal{R}(A^T) = \mathbb{R}^m, \quad \mathcal{R}(A^T) \perp \mathcal{N}(A), \tag{1.11}$$

$$\mathcal{N}(A^T) \oplus \mathcal{R}(A) = \mathbb{R}^n, \quad \mathcal{R}(A) \perp \mathcal{N}(A^T). \tag{1.12}$$

Důkaz. Důkaz převezmeme z [2, věta 1.25]. Platí

$$\mathcal{R}(A^T) \oplus \mathcal{R}(A^T)^\perp = \mathbb{R}^m.$$

Pro dokázání první části stačí ukázat, že $\mathcal{R}(A^T)^\perp \perp \mathcal{N}(A)$. Zavedeme libovolný vektor x z $\mathcal{R}(A^T)^\perp$ pro který platí následující ekvivalence

$$\begin{aligned}
x \in \mathcal{R}(A^T)^\perp &\iff \text{pro každé } y \in \mathcal{R}(A^T) \text{ platí } \langle y, x\rangle = 0\\
&\iff \text{pro každé } z \in \mathbb{R}^n \text{ platí } \left\langle A^T z, x\right\rangle = 0\\
&\iff \text{pro každé } z \in \mathbb{R}^n \text{ platí } \langle z, Ax\rangle = 0\\
&\iff Ax = 0\\
&\iff x \in \mathcal{N}(A).
\end{aligned}$$

Druhá část tvrzení vyplývá z využití prvního tvrzení na matici A^T. □

Pro matici A a symetrickou pozitivně semidefinitní matice A^TA, AA^T navíc platí následující věta.

Věta 5. *Pro obecnou obdelníkovou matici $A \in \mathbb{R}^{n\times m}$ platí*

$$\mathcal{N}(A^TA) = \mathcal{N}(A), \quad \mathcal{R}(A^TA) = \mathcal{R}(A^T),$$
$$\mathcal{N}(AA^T) = \mathcal{N}(A^T), \quad \mathcal{R}(AA^T) = \mathcal{R}(A).$$

Důkaz. Důkaz opět převezmeme z [2, lemma 5.1]. Nejprve dokážeme první rovnost. Nechť $x \in \mathcal{N}(A)$, potom platí $Ax = 0$ a také $A^TAx = 0$. Z toho plyne, že $x \in \mathcal{N}(A^TA)$. Pak tedy

$$\mathcal{N}(A) \subset \mathcal{N}(A^TA).$$

Nyní nechť $x \in \mathcal{N}(A^TA)$, potom platí $AA^TAx = 0$ a také

$$0 = x^TA^TAx = \left\langle A^TAx, x\right\rangle = \langle Ax, Ax\rangle = \|Ax\|^2.$$

Tedy i $Ax = 0$ a potom $x \in \mathcal{N}(A)$. Pak tedy

$$\mathcal{N}(A^TA) \subset \mathcal{N}(A).$$

Tím je dokázána rovnost $\mathcal{N}(A^TA) = \mathcal{N}(A)$. Rovnost pro obory hodnot, tj. $\mathcal{R}(A^TA) = \mathcal{R}(A^T)$, dostaneme přímo z rovnosti předchozí a ze vztahu (1.11).

Pro důkaz druhé sady rovnosti stačí uvažovat transponovanou matici $B = A^T$, a do první sady rovností dosadit za A i A^T. Tím ihned dostaneme vztahy $\mathcal{N}(BB^T) = \mathcal{N}(B^T)$ a $\mathcal{R}(BB^T) = \mathcal{R}(B)$. □

Podle rovnosti (1.10) a podle věty 5 dostáváme

$$\dim(\mathcal{R}(AA^T)) = \dim(\mathcal{R}(A)) = r = \dim(\mathcal{R}(A^T)) = \dim(\mathcal{R}(A^TA)), \tag{1.13}$$

kde r je hodnost matice A.

1.4 Singulární rozklad

Následující věta o singulárním rozkladu (SVD, z anglického singular value decomposition) bude pro další výklad klíčová.

Věta 6 (O singulárním rozkladu)**.** *Nechť $A \in \mathbb{R}^{n\times m}$ je obecná obdelníková matice hodnosti r. Potom existují ortogonální matice $U \in \mathbb{R}^{n\times n}$, $V \in \mathbb{R}^{m\times m}$ a matice $\Sigma \in \mathbb{R}^{n\times m}$ mající nenulové prvky pouze na diagonále tak, že platí*

$$A = U\Sigma V^T. \tag{1.14}$$

Sloupce matic U a V,

$$U = [u_1, \ldots, u_n], \quad V = [v_1, \ldots, v_m],$$

nazýváme levé respektive pravé singulární vektory. Matici Σ *lze zapsat ve tvaru*

$$\Sigma = \begin{bmatrix} \Sigma_r & 0 \\ 0 & 0 \end{bmatrix},$$

$$\Sigma_r = \begin{bmatrix} \sigma_1 & & \\ & \ddots & \\ & & \sigma_r \end{bmatrix},$$

přičemž platí

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0. \tag{1.15}$$

Nenulová čísla σ_j, $j = 1, \ldots, r$ *nazýváme singulární čísla.*

Důkaz. Protože $A^T A \in \mathbb{R}^{m \times m}$ je symetrická pozitivně semidefinitní podle věty 1 existuje ortogonální matice V a diagonální matice D tak, že platí

$$D = \operatorname{diag}(\lambda_1, \ldots \lambda_m) = V^T (A^T A) V.$$

Protože $\operatorname{rank}(A^T A) = \operatorname{rank}(A) = r$ podle (1.13) a víme, že

$$\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_r > \lambda_{r+1} = \ldots = \lambda_n = 0.$$

Tedy platí

$$A^T A v_j = v_j \lambda_j \quad \text{pro} \quad j = 1, \ldots, r$$

a navíc

$$\mathcal{N}(A^T A) = \mathcal{N}(A) = \operatorname{span}(v_{r+1}, \ldots, v_m),$$

viz (1.5), (1.7). Zřejme pro $j = 1, \ldots, r$ platí

$$\begin{aligned} A(A^T A) v_j &= A v_j \lambda_j, \\ (AA^T)(A v_j) &= (A v_j) \lambda_j, \\ AA^T w_j &= w_j \lambda_j, \end{aligned}$$

kde

$$w_j = A v_j$$

a matice $AA^T \in \mathbb{R}^{n \times n}$ je symetrická pozitivně semidefinitní s hodností r. Pro skalární součin vektorů w_j a w_k platí

$$\langle w_j, w_k \rangle = w_k^T w_j = v_k^T (A^T A v_j) = v_k^T \lambda_j v_j = \lambda_j \langle v_j, v_k \rangle,$$

tedy

$$\langle w_j, w_k \rangle = \begin{cases} 0, & j \neq k \\ \lambda_j, & j = k \end{cases}.$$

Normalizací vektorů w_j získáme

$$u_j = \frac{w_j}{\|w_j\|} = \frac{1}{\sqrt{\lambda_j}} w_j.$$

Protože

$$\dim(\mathcal{R}(AA^T)) = r$$

(viz (1.13)), pak

$$\mathcal{R}(AA^T) = \text{span}(v_1, \ldots, v_r)$$

z věty 4 vyplývá

$$\dim(\mathcal{N}(AA^T)) = n - r,$$

a dále

$$\mathcal{R}(AA^T) \perp \mathcal{N}(AA^T).$$

Pro libovolnou ortonormální bázi $\{u_{r+1}, \ldots, u_n\}$ prostoru $\mathcal{N}(AA^T)$ tedy platí, že

$$U = [u_1, \ldots, u_r, u_{r+1} \ldots, u_n]$$

je ortogonální matice, t.j. $UU^T = U^TU = I_n$. Tedy

$$D = U^T(AA^T)U.$$

Pro $j = 1, \ldots, r$ tedy platí

$$\begin{aligned} AA^Tu_j &= u_j\lambda_j, \\ A^T(AA^T)u_j &= A^Tu_j\lambda_j, \\ (A^TA)(A^Tu_j) &= (A^Tu_j)\lambda_j, \\ A^TA\widetilde{w}_j &= \widetilde{w}_j\lambda_j, \end{aligned}$$

kde

$$\widetilde{w}_j = A^Tu_j.$$

Pro skalární součin vektorů $\widetilde{w}_j$ a $\widetilde{w}_k$ platí

$$\langle \widetilde{w}_j, \widetilde{w}_k \rangle = \begin{cases} 0, & i \neq k \\ \lambda_j, & j = k \end{cases}.$$

Normalizací vektorů $\widetilde{w}_j$ získáme

$$\widetilde{v}_j = \frac{\widetilde{w}_j}{\|\widetilde{w}_j\|} = \frac{1}{\sqrt{\lambda_j}}\widetilde{w}_j.$$

Nyní zbývá ukázat, že $v_j = \widetilde{v}_j$, zřejmě

$$\widetilde{v}_j = \frac{1}{\sqrt{\lambda_j}}A^Tu_j = \frac{1}{\lambda_j}A^Tw_j = \frac{1}{\lambda_j}A^TAv_j = \frac{1}{\lambda_j}\lambda_jv_j = v_j.$$

Označíme-li

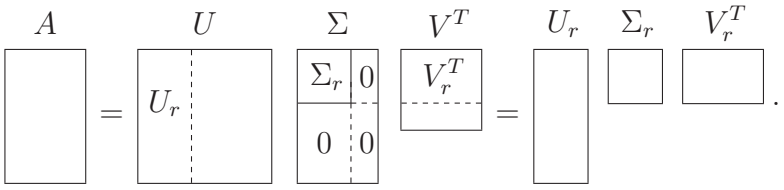
$$\sigma_j = \sqrt{\lambda_j}, \quad j = 1, \ldots, r$$

pak je důkaz hotov. □

Ukázali jsme tedy, že pro libovolnou matici A existuje singulární rozklad ve tvaru (1.14). V následující sekci ukážeme další možné tvary singulárního rozkladu a jejich využití k aproximaci matice maticí nižší hodnosti.

1.5 Aproximace matice maticí nižší hodnosti

V některých praktických úlohách je lepší použít tzv. *ekonomický tvar* singulárního rozkladu, který spočívá v odstranění některých bloků, které při násobení matic U a V^T s maticí Σ nemají vliv na výsledný součin, viz následující schéma (převzato z [2, str. 127]):



Platí tedy

$$A = U\Sigma V^T = U_r \Sigma_r V_r^T. \tag{1.16}$$

Z ekonomického tvaru je vidět, že matici A můžeme zapsat pomocí součtu vnějších součinů

$$A = \sum_{j=1}^{k} \sigma_j u_j v_j^T$$

t.j. takzvaný *dyadický rozvoj*. Následující důležitá věta popisuje aproximaci matice maticí nižší hodnosti.

Věta 7 (Schmidt, Eckart, Young, Mirsky)**.** *Nechť $A \in \mathbb{R}^{n\times m}$ je matice s hodností r. Pro libovolné přirozené číslo k, pro které platí $k < r$, matice*

$$A^{(k)} = \sum_{j=1}^{k} \sigma_j u_j v_j^T$$

je nejlepší aproximací matice A v následujícím smyslu:

$$\min_{X\in\mathbb{R}^{n\times m},\, \mathrm{rank}(X)\leq k} \|A - X\|_2 = \|A - A^{(k)}\|_2 = \sigma_{k+1}$$

a

$$\min_{X\in\mathbb{R}^{n\times m},\, \mathrm{rank}(X)\leq k} \|A - X\|_F = \|A - A^{(k)}\|_F = \sqrt{\sum_{j=k+1}^{r} \sigma_j^2}.$$

Důkaz pro jednotlivé normy lze nalézt v původních článcích [3] a [11]. Důkaz pro libovolnou ortogonálně invariantní maticovou normu lze nalézt v knize [13, str. 208–209]. Důkaz pro spektrální normu je také ve skriptech [2, str. 133, věta 5.12]. Poznamenejme, že nejlepší aproximace z předchozí věty není dána jednoznačně například když $\sigma_k = \sigma_{k+1}$.

2 Úplný problém nejmenších čtverců

V této kapitole se již budeme zabývat řešením aproximační úlohy

$$Ax \approx b \quad \text{s podmínkou} \quad b \notin \mathcal{R}(A) \tag{2.1}$$

a to pomocí úplného problému nejmenších čtverců (TLS, z anglického total least squares). K zavedení TLS budeme vycházet ze skript [2].

2.1 Motivace

Na úvod uveďme jednoduchý příklad na elektrickou vodivost, která je definovaná jako podíl elektrického proudu a napětí. Z naměřených hodnot určíme jednotlivé vodivosti viz graf na obrázku 2.1. Následně metodou nejmenších čtverců aproximujeme naměřená data přímkou.

Nejprve začneme s definicí *klasického problému nejmenších čtverců*.

Definice 3 (Klasický problém nejmenších čtverců)**.** Nechť existují $A \in \mathbb{R}^{n\times m}$ a $b \in \mathbb{R}^n$. *Klasickým problémem nejmenších čtverců* nazveme úlohu nalezení nejmenší opravy pravé strany tak, aby opravená úloha byla kompatibilní t.j.

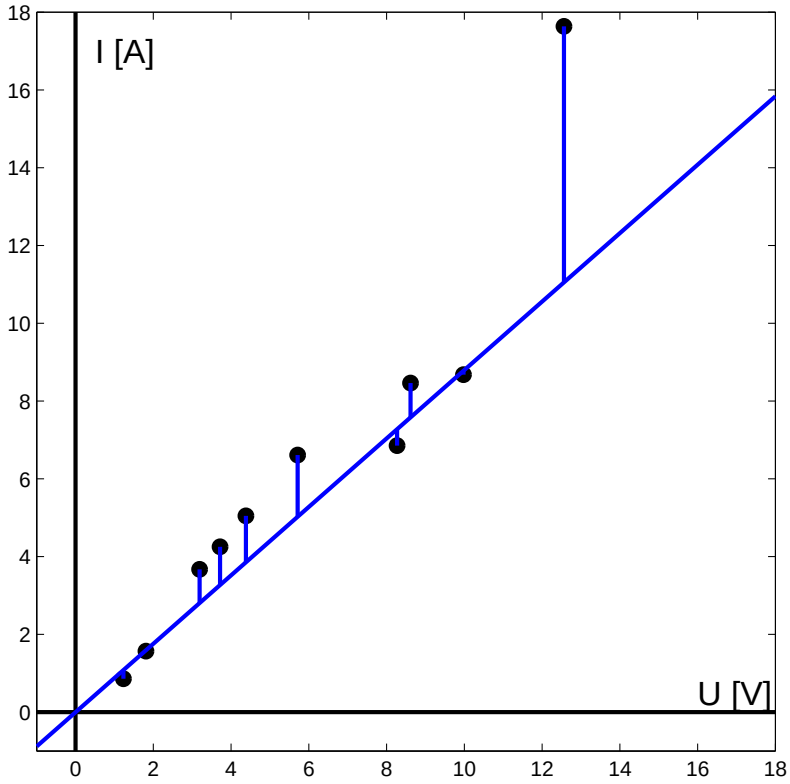
$$\min \|r\|_F \quad \text{tak, že} \quad (b+r) \in \mathcal{R}(A).$$

Pak existuje vektor $x \in \mathbb{R}^m$, který řeší opravenou úlohu. Problém lze tedy také formulovat následujícím způsobem

$$\min \|r\|_F \quad \text{tak, že} \quad Ax = b + r, \quad x \in \mathbb{R}^m.$$

Někdy se také můžeme setkat, že problém nejmenších čtverců je označován jako *lineární regrese*.

Metoda nejmenších čtverců (klasický problém je její variantou v lineárním případě; obecně $\mathcal{A}(x)$ nemusí být lineární zobrazení) je připisován C. F. Gaußovi (1777–1855), jeho základy lze nalézt v práci [7]. Ze současných učebnic zabývajících se řešením problému nejmenších čtverců odkazujeme na [10] a [1]. Z definice a také z příkladu na obrázku 2.1 na výpočet vodivosti je patrné, že v modelu je opravována pouze jedna sada naměřených dat. V příkladu na obrázku 2.1 jsou to data na ose **U** (napětí). Data na ose **I** (proudy) jsou považována za přesná. Nicméně v praxi mohou být obsaženy chyby v obou sadách naměřených dat. Tomu se budeme věnovat v následující sekci.



Obrázek 2.1: Body představují naměřené hodnoty proudu $\mathbf{I}$ (svislá osa) procházejícího rezistorem při deseti různých měřících napětích $\mathbf{U}$ (vodorovná osa). Úkolem je určit vodivost $\mathbf{G}$, resp. rezistivitu $\mathbf{R}$ součástky ($\mathbf{R} = \mathbf{G}^{-1}$). Z Ohmova zákona platí $\mathbf{UG} = \mathbf{I}$. Úloha má tedy tvar $Ax \approx b$, $A \in \mathbb{R}^{n\times m}$, $x \in \mathbb{R}^m$, $b \in \mathbb{R}^n$, kde $n = 10$, $m = 1$. Matice A obsahuje naměřené napětí, vektor pozorování b naměřené proudy, a neznámá x je vodivost. Body jsou proloženy lineární funkcí. Spojnice mezi body a funkcí ukazují odchylky měření, které se minimalizují (resp. součet jejich kvadrátů) při *lineární regresi*.

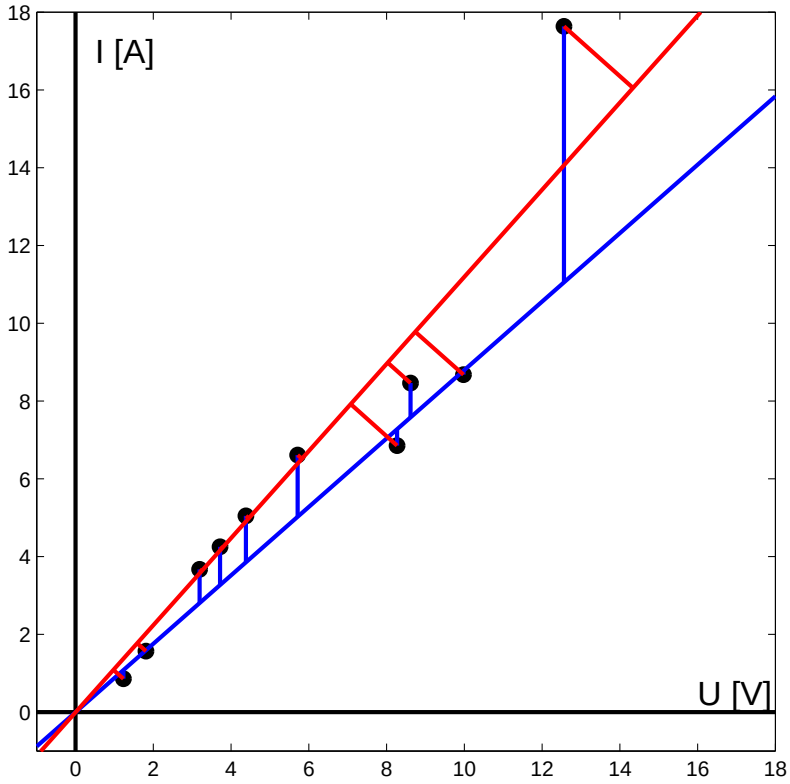
2.2 Zavedení a odvození řešení pomocí SVD

Nyní se budeme zabývat úlohou kde obě složky naměřených dat obsahují chybu, tedy budeme opravovat oboje $\mathbf{U}$ i $\mathbf{I}$, viz obrázek 2.2.
K řešení této úlohy potřebujeme následující definici.

Definice 4 (Úplný problém nejmenších čtverců)**.** Nechť existují $A \in \mathbb{R}^{n\times m}$ a $b \in \mathbb{R}^n$. *Úplným problémem nejmenších čtverců* nazveme úlohu nalezení nejmenší opravy matice A a vektoru pravé strany tak, aby opravená úloha byla kompatibilní, t.j.

$$\min \|[r, E]\|_F \quad \text{tak, že} \quad (b + r) \in \mathcal{R}(A + E).$$

Pak existuje vektor $x \in \mathbb{R}^m$, který řeší opravenou úlohu. Problém lze tedy také



Obrázek 2.2: Stejná data jako na obrázku 2.1 jsou nyní proložena lineární funkcí tak, že se minimalizuje vzdálenost bodů měřená kolmo k této funkci (resp. součet kvadrátů těchto vzdáleností). Tento přístup se nazývá *ortogonální regrese*.

formulovat následujícím způsobem

$$\min \|[r, E]\|_F \quad \text{tak, že} \quad (A + E)x = b + r, \quad x \in \mathbb{R}^m. \tag{2.2}$$

V praxi se také můžeme setkat s označením *ortogonální regrese*.

Základní učebnice týkající se úplného problému nejmenších čtverců je [18]. Pokud se vrátíme k aproximační úloze $Ax \approx b$ (2.1), kde matice A je model, b je vektor pozorování a x je vektor hledaných dat. Pak budeme opravovat matici modelu a vektor pozorování. Pro jednoduchost předpokládejme, že matice $A \in \mathbb{R}^{n\times m}$, $n > m$, má plnou sloupcovou hodnost, tedy

$$\operatorname{rank}(A) = m.$$

Dále pro vektor $b \in \mathbb{R}^n$ platí z aproximační úlohy podmínka $b \notin \mathcal{R}(A)$, pak hodnost matice A rozšířené o sloupec vektoru b vzroste o jedna, tedy

$$\operatorname{rank}([b, A]) = m + 1.$$

Pomocí SVD lze matici $[b, A]$ zapsat v dyadickém tvaru

$$[b, A] = S\Theta W^T = \sum_{j=1}^{m+1} \theta_j s_j w_j^T, \tag{2.3}$$

pro který platí

$$\theta_1 \geq \theta_2 \geq \ldots \geq \theta_m \geq \theta_{m+1} > 0. \tag{2.4}$$

Nyní budeme hledat nejmenší opravu $[r, E]$, tak aby platilo

$$\text{rank}([b, A] - [r, E]) < m + 1.$$

Podle věty 7 bude oprava tvaru

$$[r, E] = -\theta_{m+1} s_{m+1} w_{m+1}^T.$$

Pak opravená matice $[b, A]$ bude tvaru

$$[\widehat{b}, \widehat{A}] = \sum_{j=1}^{m} \theta_j s_j w_j^T.$$

V další sekci se budeme zabývat, zda opravená soustava $\widehat{A}x = \widehat{b}$ má řešení.

2.3 Existence a jednoznačnost řešení

V této sekci se budeme zabývat zda opravená soustava $\widehat{A}x = \widehat{b}$ má řešení a zda je toto řešení jednoznačné, tedy zda má původní úloha $Ax \approx b$ řešení ve smyslu TLS. Vraťme se tedy k úloze $Ax \approx b$. Jednoduchými úpravami lze úlohu zapsat ve tvaru

$$[b, A] \begin{bmatrix} -1 \\ x \end{bmatrix} \approx 0.$$

Obdobně pro opravenou soustavu dostaneme

$$[(b + r), (A + E)] \begin{bmatrix} -1 \\ x \end{bmatrix} = 0$$

respektive

$$[\widehat{b}, \widehat{A}] \begin{bmatrix} -1 \\ x \end{bmatrix} = 0. \tag{2.5}$$

Zapišme matici $[\widehat{b}, \widehat{A}]$ v jejím dyadickém tvaru

$$[\widehat{b}, \widehat{A}] = \sum_{j=1}^{m} \theta_j s_j w_j^T.$$

Nyní obě strany vynásobíme vektorem w_{m+1} a získáme

$$[\widehat{b}, \widehat{A}]w_{m+1} = \sum_{j=1}^{m} \theta_j s_j w_j^T w_{m+1} = \sum_{j=1}^{m} \theta_j s_j \langle w_{m+1}, w_j \rangle .$$

Protože sloupce matice W, tedy vektory w_j, jsou navzájem ortonormální, platí

$$[\widehat{b}, \widehat{A}]w_{m+1} = 0.$$

Uvažujme vektor w_{m+1} ve tvaru

$$w_{m+1} = \begin{bmatrix} \nu_1 \\ \nu_2 \\ \vdots \\ \nu_{m+1} \end{bmatrix}.$$

Pokud $\nu_1 \neq 0$ potom můžeme zavést pomocný vektor w'_{m+1}, který bude tvaru

$$w'_{m+1} = \frac{-1}{\nu_1} w_{m+1} = \begin{bmatrix} -1 \\ -\nu_2/\nu_1 \\ \vdots \\ -\nu_{m+1}/\nu_1 \end{bmatrix}.$$

Z porovnání s (2.5) vyplývá, že pokud $\nu_1 \neq 0$, pak původní úloha $Ax \approx b$ má řešení ve smyslu TLS ve tvaru

$$x = \frac{-1}{\nu_1} \begin{bmatrix} \nu_2 \\ \vdots \\ \nu_{m+1} \end{bmatrix} = \begin{bmatrix} -\nu_2/\nu_1 \\ \vdots \\ -\nu_{m+1}/\nu_1 \end{bmatrix}.$$

Pokud $\nu_1 = 0$ pak výše uvedený postup *zřejmě* nelze použít.

Obecně se může stát, že úloha má více než jedno (resp. nekonečně mnoho) řešení. Stačí si uvědomit, že může nastat situace $\theta_m = \theta_{m+1}$ ve (2.4) (viz poznámka na konci kapitoly 1, str. 24) a tedy lze pro konstrukci řešení použít jiný pravý singulární vektor. Může se ale také stát, že řešení vůbec nemá. Následující věta formuluje postačující podmínku pro existenci a jednoznačnost řešení TLS.

Věta 8 (Golub, Van Loan)**.** *Uvažujme matici $A \in \mathbb{R}^{n\times m}$, pro kterou platí $n > m$ a* $\operatorname{rank}(A) = m$*, t.j. A má lineárně nezávislé sloupce. Dále nechť platí, že $b \notin \mathcal{R}(A)$. Uvažujme singulární rozklady*

$$A = U\Sigma V^T \quad z \quad (1.14)\text{–}(1.15)$$

a

$$[b, A] = S\Theta W^T \quad z \quad (2.3)\text{–}(2.4).$$

Jestliže

$$\sigma_m > \theta_{m+1}, \tag{2.6}$$

pak pro matici $[b, A]$ platí, že $\nu_1 \neq 0$ a TLS problém má jednoznačné řešení.

Důkaz viz [8] případně [18]. Zde lze také nalézt diskusi týkající se případu s více než jedním řešením, pak hledáme řešení minimální v normě. Je vhodné zdůraznit, že výše uvedená věta formuluje pouze postačující podmínku nikoli nutnou.

3 Core problém

V předchozí kapitole jsme se zabývali řešením úlohy $Ax \approx b$, respektive řešením opravené úlohy soustavy $(A+E)x = b+r$. Vyslovili jsme také postačující podmínku pro existenci a vlastně také jednoznačnost řešení úlohy pomoci TLS. Zjistili jsme také, že ne vždy existuje řešení opravené úlohy pomocí TLS. Nyní se budeme zabývat metodou, která pro danou úlohu i její opravu vždy nalezne řešení, tedy pokud úloha má řešení.

3.1 Ortogonální transformace úlohy

Uvažujme opět úlohu (2.1)

$$Ax \approx b, \qquad \text{kde} \qquad A \in \mathbb{R}^{n\times m}, \quad x \in \mathbb{R}^m, \quad b \in \mathbb{R}^n. \tag{3.1}$$

Dále zaveďme ortogonální matice P a Q odpovídajících velikostí. Nyní rovnici (3.1) vynásobíme zprava P^T a, s využitím toho, že Q je ortogonální matice, t.j. $QQ^T = I$, dostaneme

$$(P^T AQ)(Q^T x) \approx P^T b. \tag{3.2}$$

Zkráceně také můžeme zapsat

$$\widetilde{A}\widetilde{x} \approx \widetilde{b}, \qquad \text{kde} \qquad \widetilde{A} = P^T AQ, \quad \widetilde{x} = Q^T x, \quad \widetilde{b} = P^T b.$$

Použijme stejnou ortogonální transformaci na opravenou úlohu (2.2), kde matice A a vektor b jsou opravené maticí E respektive vektorem r. Dostaneme

$$(P^T(A+E)Q)(Q^T x) = (P^T(b+r)). \tag{3.3}$$

Protože Frobeniova norma je ortogonálně invariantní [2], platí

$$\min \left\|[P^T r, P^T EQ]\right\|_F = \min \left\|P^T[r, EQ]\right\|_F = \min \left\|[r, E]\right\|_F .$$

Rovnici (3.3) můžeme také zapsat ve tvaru

$$(\widetilde{A} + \widetilde{E})\widetilde{x} = \widetilde{b} + \widetilde{r},$$

kde $\widetilde{E} = P^T EQ$ a $\widetilde{r} = P^T r$. Tedy hledaný vektor x bude roven

$$x = Q\widetilde{x}. \tag{3.4}$$

Předpokládejme, že matice P a Q transformují rozšířenou matici $[\widetilde{b}, \widetilde{A}]$ do tvaru

$$[\widetilde{b}, \widetilde{A}] = P^T[b, A] \begin{bmatrix} 1 & 0 \\ 0 & Q \end{bmatrix} = \left[\begin{array}{c|cc} b_1 & A_{11} & 0 \\ 0 & 0 & A_{22} \end{array}\right]. \tag{3.5}$$

Úlohu (3.2) pak můžeme přepsat ve tvaru

$$\left[\begin{array}{c|cc} b_1 & A_{11} & 0 \\ 0 & 0 & A_{22} \end{array}\right] \begin{bmatrix} -1 \\ x_1 \\ x_2 \end{bmatrix} \approx 0$$

neboli, po pronásobení

$$\begin{bmatrix} -b_1 + A_{11}x_1 \\ A_{22}x_2 \end{bmatrix} \approx 0.$$

Původní úloha se tím rozpadá na dva nezávislé podproblémy

$$A_{11}x_1 \approx b_1 \qquad \text{a} \qquad A_{22}x_2 \approx 0.$$

Vidíme, že druhý podproblém má řešení v klasickém slova smyslu (jinými slovy, je to soustava rovnic, nikoliv aproximační úloha v pravém slova smyslu). Protože zpravidla hledáme řešení minimální v normě (nemáme-li nějakou dodatečnou informaci k úloze a dobrý důvod volit řešení jiné než minimální), je nasnadě, že jako řešení druhého podproblému zvolíme $x_2 = 0$. Druhý podproblém tedy nemá vliv na řešení původní úlohy. Zbývá vyřešit první podproblém. Pokud x_1 bude řešením prvního podproblému, pak z (3.4) vidíme, že

$$x = Q\widetilde{x} = Q \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = Q \begin{bmatrix} x_1 \\ 0 \end{bmatrix}$$

bude řešením původní úlohy.

3.2 Dovětek k větě 8

Označme

$$\Sigma(M) \quad \text{a} \quad \min \Sigma(M)$$

množinu všech singulárních čísel matice M včetně násobností, resp. nejmenší singulární číslo matice M, t.j.

$$|\Sigma(M)| = \operatorname{rank}(M).$$

Ze vztahu (3.5) zřejmě platí

$$\Sigma(A) = \Sigma\left(\begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}\right) = \Sigma(A_{11}) \cup \Sigma(A_{22}), \tag{3.6}$$

proto, že P a Q jsou ortogonální matice a že množina singulárních čísel blokově diagonální matice je sjednocením množin singulárních čísel jejich diagonálních bloků. Také ale platí

$$\Sigma([b, A]) = \Sigma\left(\left[\begin{array}{c|cc} b_1 & A_{11} & 0 \\ 0 & 0 & A_{22} \end{array}\right]\right) = \Sigma([b_1, A_{11}]) \cup \Sigma(A_{22}). \tag{3.7}$$

V předpokladech věty 8 je, že A má mít lineárně nezávislé sloupce. To je splněno tehdy a jen tehdy když obě matice A_{11} a A_{22} mají lineárně nezávislé sloupce. Dále $b \notin \mathcal{R}(A)$ tehdy a jen tehdy když $b_1 \notin \mathcal{R}(A_{11})$. Ze vztahů (3.6) a (3.7) plyne následující důležitý důsledek.

Nechť A_{11} a A_{22} mají plný sloupcový rank a $b_1 \notin \mathcal{R}(A_{11})$. Pokud

$$\min \Sigma(A_{22}) < \min\{ \min \Sigma(A_{11}) , \min \Sigma([b_1, A_{11})\},$$

pak $\min \Sigma(A_{22}) = \min \Sigma(A) = \min \Sigma([b, A])$. Vidíme, že ačkoliv jsou splněny předpoklady věty 8, nerovnost (2.6) nenastane. Nejmenší singulární čísla matic A a $[b, A]$ jsou identická.

Porovnejme právě zformulovaný důsledek s tvrzeními uvedenými v úvodu textu, viz [17], [5], resp. [6]. Poznamenejme ještě, že díky tzv. prokládání singulárních čísel platí $\min \Sigma(A_{11}) \geq \min \Sigma([b_1, A_{11}])$, viz [14]. V případě core problému dokonce platí $\min \Sigma()A_{11} > \min \Sigma([b_1, A_{11}])$, viz [12], viz také věta 9 v následující sekci.

3.3 Zavedení core problému

V předchozí sekci jsme ukázali jak lze úlohy (3.1) rozložit na dva podproblémy

$$A_{11}x_1 \approx b_1 \qquad \text{a} \qquad A_{22}x_2 \approx 0.$$

V článku [12] bylo ukázáno, že ortogonální transformace ve tvaru (3.2), vedoucí na bloky tvaru (3.5) vždy existuje (může se ovšem stát, že matice A_{22} má nulový počet řádků nebo sloupců). Navíc vždy existuje taková, že matice A_{11} má minimální a A_{22} maximální dimenzi. Vzhledem k důležitosti tohoto pojmu zavedeme jeho formální definici.

Definice 5 (Core problém)**.** Řekneme, že $A_{11}x_1 \approx b_1$ je core problém v aproximační úloze (3.1) za předpokladu, že $[b_1, A_{11}]$ má minimální dimenzi (a matice A_{22} maximální dimenzi).

Nyní uveďme větu zabývající se vlastnostmi core problému.

Věta 9. *Nechť $Ax \approx b$, $b \notin \mathcal{R}(A)$ a $A_{11}x_1 \approx b_1$, kde $A_{11} \in \mathbb{R}^{\overline{n}\times\overline{m}}$, $x_1 \in \mathbb{R}^{\overline{m}}$, $b_1 \in \mathbb{R}^{\overline{n}}$ a $b_1 \notin \mathcal{R}(A_{11})$ je core problém pak*

- *Matice A_{11} má plnou sloupcovou hodnost, tedy*

$$\operatorname{rank}(A_{11}) = \overline{m}.$$

Protože $b_1 \notin \mathcal{R}(A_{11})$, pak $\overline{n} > \overline{m}$.

- *Rozšířená matice* $[b_1, A_{11}] \in \mathbb{R}^{\overline{n}\times(\overline{m}+1)}$ *má plnou řádkovou hodnost, tedy*

$$\operatorname{rank}([b_1, A_{11}]) = \overline{n},$$

  a tedy $\overline{n} = \overline{m} + 1$.

- *Obě matice* A_{11} *a* $[b_1, A_{11}]$ *mají jednoduchá singulární čísla*

$$\sigma'_1 > \sigma'_2 > \ldots > \sigma'_{\overline{m}} > 0,$$
$$\theta'_1 > \theta'_2 > \ldots > \theta'_{\overline{m}+1} > 0,$$

  pro která platí

$$\theta'_1 > \sigma'_1 > \theta'_2 > \sigma'_2 > \ldots > \sigma'_{\overline{m}} > \theta'_{\overline{m}+1}.$$

Z právě uvedené věty získáváme důležitou vlastnost core problému *nezávisle na původní úloze* $Ax \approx b$ tedy, že core problém vždy splňuje předpoklady věty 8.

Podle věty 8 má core problém $A_{11}x_1 \approx b_1$ řešení a toto řešení je právě jedno nezávisle na původní úloze $Ax \approx b$.

Je důležité upozornit, že matice A_{11} a vektor b_1 nejsou dány jednoznačně. Zřejmě pro libovolné ortogonální matice P_1 a Q_1 platí, že

$$(P_1^T A_{11} Q_1)(Q_1^T x_1) \approx P_1^T b_1$$

je opět core problém.

3.4 Tvary core problému

Core problém lze vyjádřit ve dvou základních tvarech tzv. *SVD tvar* a *bidiagonální tvar*.

3.4.1 Core problém v SVD tvaru

V předminulé sekci o ortogonální transformaci jsme pomocí ortogonálních matic P a Q transformovali původní úlohu (2.1) do tvaru (3.2). Transformací se úloha rozpadla na dva podproblémy a sice na podproblém $A_{11}x_1 \approx b_1$ minimální dimenze nazvaný core problém a podproblém $A_{22}x_2 \approx 0$ maximální dimenze. Uvažujme obecný core problém $A_{11}x_1 \approx b_1$ a SVD matice A_{11}

$$A_{11} = U_{11}\Sigma_{11}V_{11}^T.$$

Vynásobíme-li rozšířenou matici $[b_1, A_{11}]$ zleva ortogonální maticí U_{11} a zprava ortogonální maticí $\operatorname{diag}(1, V_{11})^T$, dostane tvar

$$U_{11}[b_1, A_{11}] \begin{bmatrix} 1 & 0 \\ 0 & V_{11} \end{bmatrix} = [U_{11}^T b_1 | \Sigma_{11}].$$

V článku [12] bylo ukázáno, že $U_{11}^T b_1$ má všechny složky nenulové. Tento zápis se nazývá SVD tvar core problému.

Definice 6 (SVD tvar core problému)**.** Core problém se nazývá *core problémem v SVD tvaru* pokud

$$[b_1, A_{11}] = \left[\begin{array}{c|ccc} \beta_1 & \sigma'_1 & & \\ \vdots & & \ddots & \\ & & & \sigma'_{\overline{m}} \\ \beta_{\overline{m}+1} & 0 & \dots & 0 \end{array}\right],$$

$$\begin{array}{rl} \text{kde} & \sigma'_1 > \sigma'_2 > \ldots > \sigma'_{\overline{m}} > 0 \\ \text{a} & \beta_j \neq 0, \qquad j = 1, \ldots, \overline{m}+1. \end{array}$$

3.4.2 Bidiagonální tvar core problému

Nyní se budeme zabývat bidiagonálním tvarem core problému. Pomocí ortogonálních matic P, Q lze transformovat matici $[b, A]$ do matice bidiagonálního tvaru pomocí například Householderovy reflexe. Transformace pomocí Householderových reflexí spočívá ve vytvoření posloupnosti reflexí H_1, H_2, ..., které střídavě násobíme s maticí $[b, A]$ zleva a zprava. Kde matice H_{2j-1} nulují poddiagonální prvky v j-tém sloupci rozšířené matice a matice H_{2j} nulují naddiagonální prvky v j-tém řádku systémové matice. Householderova reflexe bude vypadat

$$H_{2j-1} \ldots H_1 [b, A] \widetilde{H}_2 \ldots \widetilde{H}_{2j}$$

kde matice $\widetilde{H}_{2j}$ jsou tvaru

$$\widetilde{H}_{2j} = \begin{bmatrix} 1 & 0 \\ 0 & H_{2j} \end{bmatrix}.$$

Bidiagonalizaci ukončíme ve chvíli kdy dojde v transformaci k oddělení podproblémů, pak tedy matice A_{11} bude v bidiagonálním tvaru. Struktura matice A_{22} není důležitá. V článku [12] bylo ukázáno, že podproblém oddělený bidiagonalizací je opět core problémem, tentokrát v bidiagonálním tvaru.

Definice 7 (Bidiagonální tvar core problému)**.** Core problém se nazývá *core problémem v bidiagonálním tvaru* pokud

$$[b_1, A_{11}] = \left[\begin{array}{c|cccc} \delta_1 & \gamma_1 & & & \\ & \delta_2 & \gamma_2 & & \\ & & \ddots & \ddots & \\ & & & \delta_{\overline{m}} & \gamma_{\overline{m}} \\ & & & & \delta_{\overline{m}+1} \end{array}\right],$$

$$\begin{array}{rl} \text{kde} & \gamma_\ell > 0, \qquad \ell = 1, \ldots, \overline{m}, \\ \text{a} & \delta_j > 0, \qquad j = 1, \ldots, \overline{m}+1. \end{array}$$

4 Automatické generování aproximačních úloh s core problémem s předem známými parametry

V této kapitole si popíšeme software, který generuje core problémy $A_{11}x_1 \approx b_1$ v SVD tvaru i v bidiagonálním tvaru podle předem zadaných parametrů, resp. aproximační úlohy $Ax \approx b$ s core problémem s předem známou velikostí a s předem známými parametry. Software jsme vytvořili v prostředí programu MATLAB.

4.1 Program cpSVD. Core problém v SVD tvaru

Cílem tohoto programu bude vytvořit core problém, tedy matici A_{11} a pravou stranu b_1, v SVD tvaru. Tedy

$$[b_1, A_{11}] = \left[\begin{array}{c|ccc} \beta_1 & \sigma'_1 & & \\ \vdots & & \ddots & \\ & & & \sigma'_{\overline{m}} \\ \beta_{\overline{m}+1} & 0 & \dots & 0 \end{array}\right],$$

$$\begin{aligned} \text{kde} \quad & \sigma'_1 > \sigma'_2 > \ldots > \sigma'_{\overline{m}} > 0 \\ \text{a} \quad & \beta_j \neq 0, \qquad j = 1, \ldots, \overline{m}+1. \end{aligned}$$

Vstupními parametry programu `cpSVD` budou obecně velikost vytvořeného core problému aproximační úlohy a typ vytvoření SVD tvaru tedy způsob určení hodnot β_j a σ_j. U různých typů pak můžou přibít další parametry, které ovlivní určení hodnot β_j a σ_j.

`Program cpSVD. Zadání rozměru a typ core problému`

```
function [b_1,A_11] = cpSVD(m,typ,par)
% m        - velikost core problému
% typ = 11 - náhodná čísla beta a sigma s rovnoměrným
%            rozdělením z intervalu (0,1); příkaz rand
% typ = 12 - náhodná čísla beta a sigma s normálním
%            rozdělením N(0,1); příkaz randn
```

```
% typ = 21 - sigma lineárně rozložené v daném intervalu
%            s náhodnou perturbací; příkaz rand
% typ = 22 - dtto; příkaz randn
% typ = 31 - sigma logaritmicky roznožené v daném intervalu
%            s náhodnou perturbací; příkaz rand
% typ = 32 - dtto; příkaz randn
% typ = 40 - uživatelem zadané vektory čísel beta a sigma
% par      - parametry
```

Jednotlivé typy znamenají různé způsoby určení hodnot β_j a σ_j. Parametry v proměnné `par` se mění v závislosti na typu.

Varianta `typ == 11`**:** MATLAB pomocí příkazu `rand` určí hodnoty β_j a σ_j pseudonáhodně s rovnoměrným rozdělením z intervalu $(0, 1)$.

Typ 11. Náhodná čísla beta a sigma, rovnoměrné rozdělení

```
if typ == 11
    beta  = rand(m+1,1) - rand(m+1,1);    % vektor beta
    sigma = rand(m,1);
    sigma = sort(sigma,'descend');        % vektor sigma
end
```

Protože příkaz `rand` generuje pouze hodnoty z intervalu (0,1) tedy pouze kladná respektive nezáporná čísla a protože čísla β mohou být kladná i záporná definují se jako rozdíl dvou příkazů `rand`. Pomocí příkazu `sort` seřadí program hodnoty σ_j sestupně pro dodržení podmínky (1.15).

Varianta `typ == 12`**:** MATLAB pomocí příkazu `randn` určí hodnoty β_j a σ_j pseudonáhodně s normálním rozdělením $N(0, 1)$, tedy se střední hodnotou 0 a rozptylem 1.

Typ 12. Náhodná čísla beta a sigma, normální rozdělení

```
if typ == 12
    beta  = randn(m+1,1);                 % vektor beta
    sigma = abs(randn(m,1));
    sigma = sort(sigma,'descend');        % vektor sigma
end
```

Všimněme si, že v programu nedošlo jen k záměně příkazů `rand` a `randn`. Příkaz `randn` totiž generuje čísla s normálním rozdělením $N(0,1)$, tedy kladná i záporná, proto není zapotřebí definovat β jako rozdíl dvou příkazů, ale je třeba použít absolutní hodnotu na čísla σ, protože singulární čísla jsou kladná.

Varianta `typ == 21`: Program pomocí příkazu `rand` určí hodnoty β_j pseudonáhodně. Hodnoty σ_j určí pseudonáhodně v lineárně daném intervalu, který bude zadaný pomocí parametrů `par{1}` a `par{2}` a pomocí příkazů `linspace` a `rand`.

`Typ 21. Singulární čísla rozložená lineárně + rand`

```
if typ == 21
    beta  = rand(m+1,1) - rand(m+1,1);    % vektor beta
    sigma = abs(linspace(par{1},par{2},m)) + rand(1,m);
    sigma = sort(sigma,'descend');        % vektor sigma
end
```

Rozdílnost oproti předchozím typům program určí hodnoty σ_j pomocí příkazu `linspace`, který, v našem případě, vrátí vektor o m složkách rozložených lineárně mezi krajními hodnotami `par{1}` a `par{2}`. K takto vytvořenému vektoru navíc přičteme perturbaci vytvořenou příkazem `rand`. Díky tomu jsou čísla σ v jistém smyslu pseudonáhodná. Program navíc zajišťuje pomocí příkazu `abs` aby výsledná čísla σ byla kladná, protože uživatel může zadat jeden nebo oba parametry `par` záporné (což by ovšem neměl).

Varianta `typ == 22`: Program pomocí příkazu `randn` určí hodnoty β_j pseudonáhodně. Hodnoty σ_j určí pseudonáhodně v lineárně daném intervalu, který bude zadaný pomocí parametrů `par{1}` a `par{2}` a pomocí příkazů `linspace` a `randn`.

`Typ 22. Singulární čísla rozložená lineárně + randn`

```
if typ == 22
    beta  = randn(m+1,1);                 % vektor beta
    sigma = abs(linspace(par{1},par{2},m) + randn(1,m));
    sigma = sort(sigma,'descend')         % vektor sigma
end
```

Varianta `typ == 31`: Program určí hodnoty β_j pseudonáhodně pomocí příkazu `rand`. Hodnoty σ budou určeny pseudonáhodně v logaritmickém intervalu, který bude zadaný pomocí parametrů `par{1}` a `par{2}` a pomocí příkazů `logspace` a `rand`.

Typ 31. Singulární čísla rozložená v logaritmicky + rand

```
if typ == 31
    beta  = rand(m+1,1) - rand(m+1,1);    % vektor beta
    sigma = logspace(par{1},par{2},m).*rand(1,m);
    sigma = sort(sigma,'descend')         % vektor sigma
end
```

Rozdílnost při definování hodnot σ oproti typu 21 je v záměně příkazu `linspace` za příkaz `logspace`, který vytvoří na místo lineárního intervalu logaritmický a také, že takto vytvořený vektor násobíme perturbací vytvořenou příkazem `rand`. Všimněme si navíc, že není zapotřebí absolutních hodnot oproti typu 21, protože příkazy `logspace` a `rand` negenerují záporná čísla.

Varianta `typ == 32`**:** Oproti předchozímu typu dojde pouze ke změně z rovnoměrného na normální rozdělení. Tedy k záměně příkazu `rand` za příkaz `randn`.

Typ 32. Singulární čísla rozložená v logaritmicky + randn

```
if typ == 32
    beta  = randn(m+1,1);                 % vektor Beta
    sigma = logspace(par{1},par{2},m).*abs(randn(1,m));
    sigma = sort(sigma,'descend')         % vektor Sigma
end
```

Při definování hodnot σ_j je nutností počítat s absolutní hodnotou, protože násobením příkazů `logspace` a `randn` můžeme získat záporné σ_j, protože příkaz `randn` se řídí normálním rozdělením $N(0,1)$.

Varianta `typ == 40`**:** Program vloží do vektorů β a σ hodnoty z parametru `par`.

Typ 40. Zadané vektory čísel beta s sigma

```
if typ == 40
    beta  = par{1};                       % vektor beta
    sigma = abs(par{2});                  % vektor sigma
    if length(beta) ~= m+1
        disp('CHYBA: špatná délka vektoru beta')
        break
    end
    if length(sigma) ~= m
        disp('CHYBA: špatná délka vektoru sigma')
        break
```

```
    end
    [sigma,idx] = sort(sigma,'descend');
    beta(1:m+1) = beta(idx);
end
```

V programu je pro jistotu provedena kontrola zda uživatel zadal vektory β a σ správné délky. Dále je provedeno setřídění matice $[b_1, A_{11}]$ po řádcích podle velikosti singulárních čísel σ_j.

Další částí programu je kontrola podmínky jednoduchosti singulárních čísel

$$\sigma_1 > \sigma_2 > \ldots > \sigma_{\overline{m}} > 0,$$

tedy zajistit, aby singulární čísla byla *různá*. Program bude porovnávat sousední singulární čísla σ_j. K tomu je zapotřebí jednoduchý `for` cyklus doplněný o podmínku `if`. Při nalezení stejných singulárních čísel program singulární číslo s nižším indexem nepatrně zvětší vynásobí ho hodnotou $(1 + \varepsilon/2)$ a singulární číslo s větším indexem nepatrně zmenší vynásobí ho hodnotou $(1 - \varepsilon/2)$.

Program cpSVD, kontrola jednoduchosti čísel sigma

```
eps = 0.001;                    % hodnota epsilon
for j=1:m-1                     % kontrola s_j je ruzne s_{j+1}
    if sigma(j)-sigma(j+1) <= eps
        sigma(j)   = sigma(j  )*(1+eps/2);
        sigma(j+1) = sigma(j+1)*(1-eps/2);
    end
end
```

Protože nikdy nenastane situace, že rozdíl dvou sousedních čísel σ_j bude roven nule požadujeme, aby jejich rozdíl byl větší než epsilon. Parametr ε (`eps`) bude volitelný z uživatelského rozhraní, nyní je nastaven na hodnotu 10^{-3}.

Závěrečná část programu sestaví vzniklé vektory β a σ do matice v SVD tvaru core problému.

Program cpSVD. Sestavení matice

```
b_1  = beta(:);
A_11 = [ diag(sigma) ; zeros(1,m) ];
```

Všimněme si, že příkaz `beta(:)` zajistí, že v návratové proměnné b_1 bude vždy sloupcový vektor.

4.2 Program cpBDG. Core problém v bidiagonálním tvaru

Cílem tohoto programu bude, aby vytvořil matici v Bidiagonálním tvaru včetně podmínek pro Bidiagonální tvar. Tedy

$$[b_1, A_{11}] = \left[\begin{array}{c|cccc} \delta_1 & \gamma_1 & & & \\ & \delta_2 & \gamma_2 & & \\ & & \ddots & \ddots & \\ & & & \delta_{\overline{m}} & \gamma_{\overline{m}} \\ & & & & \delta_{\overline{m}+1} \end{array}\right],$$

$$\begin{array}{rll} \text{kde} & \gamma_\ell > 0, & \ell = 1, \ldots, \overline{m}, \\ \text{a} & \delta_j > 0, & j = 1, \ldots, \overline{m}+1. \end{array}$$

Vstupními parametry programu `cpBDG` budou obecně velikost vytvořeného core problému aproximační úlohy a typ vytvoření Bidiagonálního tvaru, tedy způsob určení hodnot δ_j a γ_j. U různých typů pak můžou přibít další parametry, které ovlivní určení hodnot δ_j a γ_j.

`Program cpBDG. Zadánírozměru a typ core problému`

```
function [b_1,A_11] = cpBDG(m,typ,par)
% m        - velikost core problému
% typ = 11 - nahodná čísla delta a gamma s rovnoměrným
%            rozdělením z intervalu (0,1); příkaz rand
% typ = 12 - nahodná čísla delta a gamma s normálním
%            rozdělením N(0,1); příkaz randn
% typ = 21 - delta a gamma lineárně rozložená v daném
%            intervalu s náhodnou perturbací; příkaz rand
% typ = 22 - dtto; příkaz randn
% typ = 31 - delta a gamma logaritmicky rozložená v daném
%            intervalu s náhodnou perturbací; příkaz rand
% typ = 32 - dtto; příkaz randn
% typ = 40 - uživatelem zadané vektory čísel delta a gamma
% par      - parametry
```

Jednotlivé typy různě určí hodnoty δ_j a γ_j. Parametry v proměnné `par` se mění v závislosti na `typu`.

Varianta `typ == 11`: Program určí hodnoty δ a γ pseudonáhodně s rovnoměrným rozdělením z intervalu $(0, 1)$ pomocí příkazu `rand`.

Typ 11. Náhodná čísla delta a gamma, rovnoměrné rozdělení

```
if typ == 11
    delta = rand(m+1,1);   % vektor delta
    gamma = rand(m,1);     % vektor gamma
end
```

Příkaz rand generuje hodnoty z intervalu $(0, 1)$, tedy je zajištěna podmínka, že čísla δ_j a γ_j jsou kladná.

Varianta `typ == 12`**:** Program určí hodnoty δ a γ pseudonáhodně s normálním rozdělením $N(0, 1)$ pomocí příkazu `randn`.

Typ 12. Náhodná čísla delta a gamma, normální rozdělení

```
if typ == 12
    delta = abs(randn(m+1,1));  % vektor delta
    gamma = abs(randn(m,1));    % vektor gamma
end
```

Rozdílnost mezi typy 11 a 12 není pouze v příkazu `randn`, ale protože příkaz `randn` generuje kladné i záporné hodnoty je nutné použít absolutní hodnotu na vytvořená čísla δ_j a γ_j pro dodržení podmínky, že hodnoty jsou kladné.

Varianta `typ == 21`**:** Program určí hodnoty δ a γ pseudonáhodně v lineárně daném intervalu, který bude zadaný pomocí parametru `par` a pomocí příkazu `linspace` a `rand`.

Typ 21. Delta a gamma rozložena lineárně + rand

```
if typ == 21
    delta = abs(linspace(par{1},par{2},m+1) + rand(1,m+1);
    gamma = abs(linspace(par{1},par{2},m) + rand(1,m);
end
```

Hodnoty δ a γ jsou generovány podobně jako hodnoty σ pro typ 21 v předchozí sekci. Stejně tak u typ; 22, 31 a 32.

Varianta `typ == 22`**:** Program určí hodnoty δ a γ pseudonáhodně v lineárně daném intervalu, který bude zadaný pomocí parametru `par` a pomocí příkazu `linspace` a `randn`.

Typ 22. Delta a gamma rozložená lineárně + randn

```
if typ == 22
    delta = abs(linspace(par{1},par{2},m+1) + randn(1,m+1));
    gamma = abs(linspace(par{1},par{2},m) + randn(1,m));
end
```

Varianta `typ == 31`**:** Program určí hodnoty δ a γ pseudonáhodně v logaritmicky daném intervalu, který bude zadaný pomocí parametru `par` a pomocí příkazu `logspace` a `rand`.

Typ 31. Delta a gamma rozložená logaritmicky + rand

```
if typ == 31
    delta = logspace(par{1},par{2},m+1) + rand(1,m+1);
    gamma = logspace(par{1},par{2},m) + rand(1,m);
end
```

Varianta `typ == 32`**:** Program určí hodnoty δ a γ pseudonáhodně v logaritmicky daném intervalu, který bude zadaný pomocí parametru `par` a pomocí příkazu `logspace` a `randn`.

Typ 32. Delta a gamma rozložená logaritmicky + randn

```
if typ == 32
    delta = logspace(par{1},par{2},m+1) + abs(randn(1,m+1));
    gamma = logspace(par{1},par{2},m) + abs(randn(1,m));
end
```

Varianta `typ == 40`**:** Podobně jako v případě programu `cpSVD`, i zde při volbě tohoto typu program vloží do vektorů δ a γ hodnoty zadané přímo uživatelem obsažené v proměnné `par`.

Typ 40. Zadané vektory čísel delta a gamma

```
if typ == 40
    delta = par{1};                % vektor delta
    gamma = par{2};                % vektor gamma
    if length(delta) ~= m+1
```

```
        disp('CHYBA: špatná délka vektoru delta')
        break
    end
    if length(gamma) ~= m
        disp('CHYBA: špatná délka vektoru gamma')
        break
    end
end
```

Pro jistotu program provede kontrolu zda uživatel zadal vektory δ a γ se správnou délkou. Závěrečná část programu vytvoří z vektorů δ a γ matici $[b_1, A_{11}]$ v bidiagonálním tvaru core problému.

Program cpBDG, vytvoření matice

```
M    = diag(delta) + diag(beta,1);
b_1  = M(:,1);
A_11 = M(:,2:m+1);
```

4.3 Program approxpb. Generátor aproximačních úloh s core problémem

Cílem je vytvořit program, který generuje aproximační úlohu $Ax \approx b$ s core problémem $A_{11}x_1 \approx b_1$ s předem zvolenou velikostí matice A a zvoleným tvarem, velikostí a způsobem vytvoření core problému. Programu `approxpb` využije ke generován core problémů programy `cpSVD` a `cpBDG`. Ty vytvoří core problém ve tvaru SVD respektive v bidiagonálním tvaru. Těmto programům jsme se detailněji věnovali v minulých sekcích.

Vstupními parametry programu `approxpb` budou velikost matice A, velikost core problému a dále volba tvaru core problému, typ vytvoření zvoleného core problému a případně další parametry potřebné při tvorbě tvaru core problému.

Program approxpb. Vstupní a výstupní parametry.

```
function [b,A,b_1,A_11,A_22,P,Q] = approxpb(M,N,m,tvar,typ,par)
% tvar = 1  - core problém v SVD tvaru
% tvar = 2  - core problém v Bigiagonálním tvaru
% P,Q       - ortogonální matice
```

V další části program vytvoří core problém ve tvaru, který zvolí uživatel. Pro tvar roven 1 program s pomocí programu `cpSVD` vytvoří core problém v SVD tvaru. Pro hodnotu 2 vytvoří s pomocí programu `cpBDG` core problém v bidiagonálním tvaru.

Program approxpb. Volba tvaru core problému

```
if tvar == 1   % core problém v SVD tvaru
    [b_1,A_11] = cpSVD(m,typ,par);
end
if tvar == 2   % core problém v bidiagonálním tvaru
    [b_1,A_11] = cpBDG(m,typ,par);
end
```

Volba parametru `typ` určí s jakým rozdělením bude program generovat matici A_{22}. Pro typy, které jsou modulo 10 menší než 2 bude program používat rovnoměrné rozdělení v intervalu $(0, 1)$ s příkazem `rand`. Pro typy, které jsou modulo 10 rovné 2 bude program používat normální rozdělení $N(0, 1)$. Dále program zmenší singulární čísla matice A_{22} na základě velikosti nejmenšího singulárního čísla matice $[b_1, A_{11}]$ a zvoleného parametru *defect* jenž bude volitelný z uživatelského rozhraní a jehož velikost je nyní nastavena na 10^{-2}.

Program approxpb. Generování matice A_{22}

```
if mod(typ,10) < 2
    A_22 = rand(N-m-1,M-m);     % rovnoměrné rozdělení
end
if mod(typ,10) == 2
    A_22 = randn(N-m-1,M-m);    % normální rozdělení
end
S1 = min(svd([b_1,A_11]));  % min. sing. číslo matice [b_1,A_11]
S2 = min(svd(A_22));        % min. sing. číslo matice A_22
defect = 0.01;              % defekt matice A_22
A_22   = A_22*S1/S2*defect;
```

Závěrečná část programu sestrojí vektor pravé strany b, který vznikne z vektoru b_1 vytvořeném programem `cpSVD` respektive `cpBDG` a doplněném o nuly. Matice A vznikne jako blokově diagonální matice s bloky matic A_{11}, která je vytvořena programem `cpSVD` respektive `cpBDG` a A_{22}. Na závěr program provede pomocí ortogonálních matic P a Q ortogonální transformaci vektoru b a matice A.

Program approxpb. Sestavení matice A a vektoru pravé strany b

```
b = [b_1;zeros(N-m-1,1)];        % sestavení pravé strany
A = blkdiag(A_11,A_22);          % sestavení matice

P    = orth(rand(N));            % náhodné ortogonální matice
Q    = orth(rand(M));

b = P*b;                         % ortogonální transformace
A = P*A*Q';
```

Všimněme si, že některé parametry jako `typ` nebo `m` zadávané do programu `approxpb` jsou zároveň parametry pro programy `cpSVD`, respektive `cpBDG`, které byly popsány v předchozích sekcích.

4.4 Databáze náhodných aproximačních úloh s core problémem stejných vlastností

Smyslem programového balíku `approxpb`, `cpSVD` a `cpBDG` je vytvoření několika databází náhodných aproximačních úloh. Každá databáze by měla obsahovat několik tisíc problémů stejných rozměrů obsahujících core problém charakterizovaný stejnými parametry (rozměry, distribuce singulární čísel σ_ℓ a čísel β_j, resp čísel γ_ℓ a δ_j). Jak již bylo řečeno v úvodu, tyto testovací databáze by měly rozluštit otázku, zda lze cíleně generovat problémy $Ax \approx b$, jejíž prvky se budou chovat jako nezávislé, identicky distribuované náhodné proměnné a navíc matice A bude mít lineárně nezávislé sloupce a přitom aproximační problém nebude mít řešení ve smyslu TLS. Nebo že nelze cíleně generovat aproximační problémy a je možné statistickými metodami odhalit existenci core problému uvnitř aproximační úlohy.

Závěr

Cílem této bakalářské práce bylo nastudovat teorii úplného problému nejmenších čtverců (TLS) pro řešení aproximačních úloh $Ax \approx b$. Zejména seznámit se s tzv. teorií core problému. Hlavním cílem pak bylo vytvoření programu (sady rutin) vytvořené v prostředí programu MATLAB, který bude umět generovat core problémy v různých velikostech a různých typů, resp. který bude umět generovat aproximační úlohu $Ax \approx b$ předepsaných rozměrů s (netriviálním) core problémem předepsaných vlastností. Smyslem tohoto programu je vytvoření nástroje pro generování testovacích sad úloh, na nichž se budou testovat některé statistické hypotézy.

Před samotným zavedením core problému bylo třeba objasnit několik důležitých pojmů. Jako první jsme zavedli singulární rozklad (SVD), k jehož zavedení jsme využili spektrálního rozkladu symetrických pozitivně (semi)definitních matic. Ukázali jsme, že takový spektrální rozklad je vždy reálný (tj. vlastní čísla i vlastní vektory mají nulové imaginární složky). V návaznosti na to jsme také ukázali, že i singulární rozklad reálné matice lze vždy zkonstruovat reálný. Dále jsme uvedli důležitou větu (Schmidtovu–Eckartovu–Youngovu–Mirského) o aproximaci matice maticí nižší hodnosti.

Výše zmíněnou větu jsme využili v analýze úplného problému nejmenších čtverců. Také jsme zde zmínili postačující (nikoliv nutnou) podmínku pro existenci TLS řešení daného aproximačního problému, tzv. Golubovu–Van Loanovu podmínku. Speciálně jsme zjistili, že ne každý aproximační problém má vždy řešení ve smyslu TLS.

Ukázali jsme, že ortogonální transformací se původní aproximační problém může rozpadnout na dva podproblémy z niž první, je-li minimální dimenze, se nazývá *core problém* a druhý, existuje-li, má vždy triviální řešení. Zjistili jsme tedy, že druhý problém nemá vliv na řešení úlohy. Následně jsme uvedli větu s vlastnostmi core problému, ze kterých vyplynulo, že core problém vždy splňuje výše zmíněnou Golubovu–Van Loanovu postačující podmínku pro existenci TLS řešení. Tak jsme se dostali k nejdůležitějšímu poznatku: pokud má původní aproximační úloha TLS řešení, můžeme ho vždy nalézet pomocí core problému.

Jak už bylo zmíněno, práce by měla být námětem pro další zkoumání zejména statistických vlastností aproximačních úloh a jejich případné korelaci s tím, zda úloha má nebo nemá TLS řešení. Pomocí vytvořeného programu lze konstruovat např. aproximační úlohy s maticí A plné sloupcové hodnosti, které ale nemají řešení ve smyslu TLS. Navíc neexistence řešení nebude na první pohled viditelná (struktura core problému nebude viditelná). Otázkou je, zda bude viditelná na pohled druhý, např. s použitím některých vybraných statistických metod.

Literatura

[1] A. Björck: *Numerical methods for least squares problems*, SIAM Publishing, Philadelphia, 1996.

[2] J. Duintjer Tebbens, I. Hnětynková, M. Plešinger, Z. Strakoš, P. Tichý: *Analýza metod pro maticové výpočty: Základní metody*, Matfyzpress, Praha, 2012.

[3] C. Eckart, G. Young: *The approximation of one matrix by another of lower rank*, Psychometrica 1(3) (1936), str. 211–218.

[4] M. Fiedler: *Speciální matice a jejich použtí v numerické matematice*, SNTL, Státní nakladatelství technické literatury (edice TKI, Teoretická knižnice inženýra), Praha, 1981.

[5] J. Franc: *Robustified total least squares*, v Doktorandské dny 2010, sborník workshopu doktorandů FJFI oboru Matematické inženýrství, P. Ambrož, Z. Masáková (editoři), ČVUT, Praha, 2010, str. 47–58. Dostupné na: `http://kmwww.fjfi.cvut.cz/ddny/historie/10-sbornik.pdf`.

[6] J. Franc: *Some computational aspects of robustified total least squares*, v SPMS 2011 — Stochastic and Physical Monitoring Systems, Proceedings of the international conference, T. Hobza (editor), ČVUT, Praha, 2011, str. 35–48. Dostupné na: `http://gams.fjfi.cvut.cz/SPMS/2011/SPMS_2011_Proceedings.pdf`.

[7] C. F. Gauß: *Theoria motus corporum coelestium in sectionibus conicis Solem ambientium*, Perthes et Besser, Hamburg, 1809.

[8] G. H. Golub, C. F. Van Loan: *An analysis of the total least squares problem*, SIAM Journal on Numerical Analysis 17(6) (1980), str. 883–893.

[9] G. H. Golub, C. F. Van Loan: *Matrix computations* (čtvrté vydání), The Johns Hopkins University Press, Baltimore, 2013.

[10] C. L. Lawson, R. J. Hanson: *Solving least squares problems*, SIAM Publishing, Philadelphia, 1987.

[11] L. Mirsky: *Symmetric gauge functions and unitarily invariant norms*, The Quarterly Journal of Mathematics 11(1) (1960), str. 50–59.

[12] C. C. Paige, Z. Strakoš: *Core problems in linear algebraic systems*, SIAM Journal on Matrix Analysis and Applications 27(3) (2005), str. 861–875.

[13] G. W. Stewart, J.-G. Sun: *Matrix perturbation theory*, Academic Press, San Diego and London, 1990.

[14] R. C. Thompson: *Principal submatrices IX: Interlacing inequalities for singular values of submatrices*, Linear Algebra and its Applications 5(1) (1972), str. 1–12.

[15] S. Van Huffel: *Analysis of the total least squares problem and its use in parameter estimation*, PhD Thesis, KU Leuven, Leuven, 1987.

[16] S. Van Huffel: *Total least squares and errors-in-variables modeling: Bridging the gap between statistics, computational mathematics and engineering.* Preprint ESAT-SISTA/TR 2004-44. Dostupné na: `ftp://ftp.esat.kuleuven.be/sista/vanhuffel/reports/04-44.ps.gz`.

[17] S. Van Huffel: *Total least squares and errors-in-variables modeling: Bridging the gap between statistics, computational mathematics and engineering*, v COMPSTAT 2004 — Proceedings in Computational Statistics, J. Antoch (editor), Springer Verlag, 2004, str. 539–555.

[18] S. Van Huffel, J. Vadewalle: *The total least square problem: Computational aspect and analysis*, SIAM Publishing, Philadelphia, 1991.

[19] S. Van Huffel (editor): *Recent advances in total least squares and errors-in-variables modeling*, SIAM Publishing, Philadelphia, 1997.

[20] S. Van Huffel, P. Lemmerling (editoři): *Total least squares and errors-in-variables modeling. Analysis, algorithms and applications*, Kluwer Academic Publishers, Dordrecht, 2002.

[21] D. S. Watkins: *Fundamentals of matrix computations* (třetí vydání), John Wiley & Sons Inc., New York, 2010.